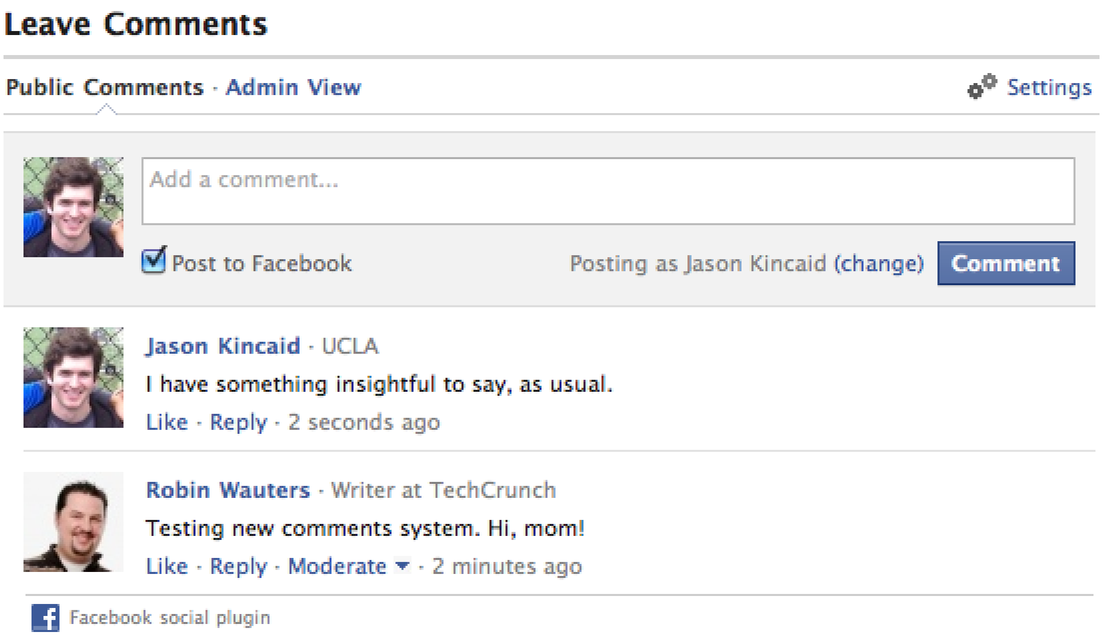

Leave Comments

Public Comments · Admin View Settings

Add a comment...

☑ Post to Facebook Posting as Jason Kincaid (change) Comment

Jason Kincaid · UCLA
I have something insightful to say, as usual.
Like · Reply · 2 seconds ago

Robin Wauters · Writer at TechCrunch
Testing new comments system. Hi, mom!
Like · Reply · Moderate ▾ · 2 minutes ago

Facebook social plugin

Predicting the Number of Facebook Comments Within K Hours

Prepared for CISC-451 | Supervised Learning Competition Two

Tao Shan | Adam Qian | Hansen Liu

Software packages & Link:

Python 3.10.2: https://www.python.org/
Pandas: https://pandas.pydata.org/getting_started.html
Sci-kit learn: https://scikit-learn.org/stable/getting_started.html
Seaborn https://seaborn.pydata.org/installing.html
Matplotlib https://matplotlib.org/stable/
[All additional relevant packages are also imported from our master notebook.]

Datasets:
Provided C2T2_Train .csv & C2T2_Test.csv data (Testing was only used to produce final 'C2T2_Test_Lableled.csv', bulk of work done with Training data due to labeled data).

Analytics Process

Exploration (Starting at In [2])

All plots are in .png files or in code output.

We first determined that our data has 129,999 records, 55 features, no missing values, and is fully numerical. Columns with 0-5, 5-50, and >50 unique values were grouped and visualized, and we noticed some columns having a disproportionately small number of unique values.

We removed one in every pair of features that had a correlation coefficient greater than 0.95.

Boxplot outlier visualizations are provided in code. We found that 10-20% of the records for each continuous numeric variable could be considered outliers using IQR.

Preparation

We decided to manually calculate metric Weight of Evidence (WOE) and Information Value (IV) to try to rank features by importance. The WOE process helps to transform our continuous independent variables into bins based on similarity.

We created a version of our dataset that was normalized using RobustScaler. This will reduce the impact of outliers on our results.

The attribute “PageCategory” is a nominal categorical attribute, so we converted it from numerical to strong to adjust for this. We also used a HashEncoder to reduce the number of unique values from 70 to 6 to potentially increase the predictive power of the attribute. We removed the “ID” column because it is not relevant for prediction.

Modeling (Starting at In [46])

We started modeling with Random Forest, the most intuitive choice for this problem. We have ultimately settled upon the XGB Regressor and the Gradient Booster because of their superior performance.

Given that the “PredictAfterHrs” feature was 24 hours for 98% of our data, we found that building 2 separate models yielded better performance. The first model handles records with a

“PredictAfterHrs” equal to 24, and the other handles records that are not 24. The model types are as follows:

- XGB Regressor (To predict when ‘PredictionAfterHrs’ = 24)
- Gradient Booster(To predict when ‘PredictionAfterHrs’ != 24)

We created these two models based on the feature “PredictAfterHrs” because not doing so would severely hinder any model’s ability to predict a record with “PredictAfterHrs” not equal to 24. Because this feature contains few unique values, any regressor or classifier would artificially devalue its importance. In reality, “PredictAfterHrs” is the fulcrum of this prediction problem and must be represented as such for any generalizable solution.

Comprehensive model testing, RMSE reporting, hyper-pruning parameter testing and analysis included in code.

Considered approaches:

- *Building 2 separate models: one for hours 1-23 (Boosting and bagging due to the significantly less amount of data) and the other for only hours 24*
- *Using Comments/Hour as the new target attribute (calculated by ‘CommentsNumber’/’PredictAfterHrs’)*
- *Stratified sampling to ensure the model includes an adequate amount of data from hours 1-23*
- *Not doing anything special at all*

Modeling Cont.

Early stopping for Gradient Booster (“PredictAfterHrs” !=24) may prevent overfitting. IN[329] is a good example of a potentially overfit model, as any models with a large discrepancy between accuracy from training and testing data are good candidates of overfitting.

Final Accuracies

Our submission is a concatenated column (from both models). Our best model had an RMSE of 15.349 for the validation set and 15.049 for the training dataset - there is no sign of overfitting or underfitting.

Bonus

To convert this into a classification problem, we discretize the target attribute “CommentsNumber” by binning it.
CommentsNumber = 0: no comments, 73102 labels in training data
CommentsNumber = 1-10: has low comments, 42295 labels in training data
CommentsNumber = 11-50: has some comments, 10605 labels in training data
CommentsNumber > 50: high comment numbers, 3997 labels in training data
We built 3 models, Random forest, Xgboost and Gradient boost. Random forest has the highest accuracy score(0.835) and ROC score(0.958)